Authentic 2 - Development #42639

ajouter un index trigram sur CONCAT(first_name, ' ', last_name)

07 mai 2020 18:28 - Benjamin Dauvergne

Statut:	Fermé	**Début:**	07 mai 2020
Priorité:	Normal	**Echéance:**	
Assigné à:	Valentin Deniaud	**% réalisé:**	0%
Catégorie:		**Temps estimé:**	0:00 heure
Version cible:			
Patch proposed:	Non	**Planning:**	Non

Description

Le but serait de permettre une recherche approximative sur le nom via l'API (via un paramètre ?q=).

Demandes liées:

Lié à Authentic 2 - Development #45419: détection homonymes/doublons à la cré...	**Fermé**	**23 juillet 2020**

Historique

#2 - 07 mai 2020 18:29 - Benjamin Dauvergne

- Projet changé de Nord - CD59 à Authentic 2

#3 - 07 mai 2020 23:11 - Thomas Noël

Ne pourrait-on pas généraliser aux attributes avec searchable=True ?

#4 - 08 mai 2020 13:29 - Benjamin Dauvergne

Les index trigram coûtent cher à maintenir, à chercher et les résultats sont complexes à trier donc je ne sais pas trop. L'autre problème c'est la recherche full-text sur trigram, ça n'existe pas vraiment il faut créer une formule pour décomposer le texte et dire dans quel champ on cherche, exemple sur zoo/nanterre initialement, si je recherche "Jean Marc Dupond 01/09/1975" :

- déjà je sépare les mots de ce qui semble être une date, donc ["Jean", "Marc", "Dupond"] / datetime(1975, 9, 1)
- ensuite je découpe la suite de mots en groupes que je match alternative sur le nom ou le prénom:
- ou (prénom="jean" et nom="marc dupond") ou (nom="jean" et prénom="marc dupond")
  ou (prénom="jean marc" et nom="dupond") ou (nom="jean marc" et prénom="dupond")
  ou (nom="jean marc dupond")
- ensuite pour chaque cas il faut donner un score et le combiner avec le score sur la date (la date c'est 0 ou 1 c'est assez simple), genre (score_prenom*score_nom) ** (2 - score_date) > 0.5

Très vite c'est devenu chiant de faire comme ça et au lieu d'avoir deux index trigrammes sur nom et prénom j'ai fait un index sur les deux concaténés CONCAT(prenoms || ' ' || nom) me donnant une seule clause 'jean marc dupond' ~ CONCAT(prenoms || ' ' || nom) beaucoup moins coûteuse pour postgres.

Si on commence à rajouter des champs autres ça devient impossible à gérer en mode "recherche full text" parce qu'on ne sait pas comment classer les résultats; si on peut rester sur nom/prénom/date de naissance c'est plus simple; et peut-être téléphone et email, l'important c'est qu'on arrive à extraire les éléments de la chaîne recherchée.

#5 - 08 mai 2020 14:57 - Thomas Noël

Effectivement, je comprends la difficulté, ceci dit ça ne serait pas logique d'avoir un boolean "searchable" sur certains attributs et de ne plus le prendre en charge (on va vite aller vers des incompréhensions).

Avec un index trigramme limité à prénom et nom, une idée pourrait être que la recherche devienne un mix de cet index avec free_text_search ? Ceci étant je n'ai pas d'idée précise sur les pondérations/priorisations à faire... Et peut-être faudrait-il être un peu plus "rusé", ne chercher dans les mails que si un terme contient "@", dans les dates que si un terme contient un "/".

En tout cas, en dehors de ce travail à faire sur l'usager de l'index trigramme, je pense que sa création s'impose, et comme c'est l'objet de ce ticket, j'arrête de trop dériver ici ;)

#6 - 08 mai 2020 23:14 - Benjamin Dauvergne

Thomas Noël a écrit :

> Effectivement, je comprends la difficulté, ceci dit ça ne serait pas logique d'avoir un boolean "searchable" sur certains attributs et de ne plus le prendre en charge (on va vite aller vers des incompréhensions).
>
> Avec un index trigramme limité à prénom et nom, une idée pourrait être que la recherche devienne un mix de cet index avec free_text_search ?

> Ceci étant je n'ai pas d'idée précise sur les pondérations/priorisations à faire... Et peut-être faudrait-il être un peu plus "rusé", ne chercher dans les mails que si un terme contient "@", dans les dates que si un terme contient un "/".

C'est exactement ce que je décris pour les mails et les dates mais peut-être mal.

Pour les autres champs on pourra chercher dedans les termes qui ne sont ni mails ni des dates comme on le fait maintenant, i.e. un terme à la fois mais c'est vrai que je serai plus à l'aise si on demandait un préfixe dans ce cas (comme sur le RSU on demande le préfixe '#' pour cherche dans les numéros de RSU ou le clé de fédération des logiciels métiers), genre pou champ 'rsa' contenant un identifiant de dossier RSA, taper « rsa:1234 ». Ce qui ne résout pas le problème de la pondération par rapport à la recherche approximative sachant que j'ai du mal à voir l'intérêt de combiner les deux en fait. Ça fait un joli exercice théorique mais ça n'a pas vraiment d'utilité (si tu as un identifiant unique style numéro de téléphone ou email, à la rigueur tu fais une typo et avoir une recherche approximative est intéressante, mais chercher avec plusieurs identifiants n'a pas forcément d'intérêt).

#7 - 09 mai 2020 02:23 - Thomas Noël

En fait mon seul objectif ici est que le fait de cocher la case "Pris en compte dans les recherches :" (dans hobo, qui devient le searchable=True dans A2) garde sa fonction. Ce qui serait super c'est que la recherche trigram joue sur prénom et nom (avec plus d'intelligence que ce qu'on fait actuellement, donc). Et que pour les autres champs, on arrive à faire des recherches aussi. Et mixer les deux, quitte à décider que ce sont les prénom/nom qui gagnent.

Je ne sais pas si je suis assez clair. Peut-être qu'on dit la même chose :)

#8 - 11 mai 2020 11:09 - Benjamin Dauvergne

Thomas Noël a écrit :

> En fait mon seul objectif ici est que le fait de cocher la case "Pris en compte dans les recherches :" (dans hobo, qui devient le searchable=True dans A2) garde sa fonction. Ce qui serait super c'est que la recherche trigram joue sur prénom et nom (avec plus d'intelligence que ce qu'on fait actuellement, donc). Et que pour les autres champs, on arrive à faire des recherches aussi. Et mixer les deux, quitte à décider que ce sont les prénom/nom qui gagnent.
>
> Je ne sais pas si je suis assez clair. Peut-être qu'on dit la même chose :)

Nouveau plan :

- plutôt que d'attribuer un comportement particulier à nom/prénom, construire un champ display_name comme dans w.c.s. à partir de nom/prénom (ce sera configurable comme w.c.s.) ce champ est particulier et indexé via trigram/unaccent/lower (zoo-like);
- pour les tous les autres champs marqué "Pris en compte dans les recherches", créer une colonne fts comme dans w.c.s et l'alimenté, la recherche full text se fera dessus.

#9 - 24 juillet 2020 12:27 - Frédéric Péters

- Lié à Development #45419: détection homonymes/doublons à la création d'un utilisateur ajouté

#10 - 15 septembre 2020 17:36 - Valentin Deniaud

J'ai dit que je regardais #46424 et je me demande si traiter ce ticket est un prérequis. Moi je vais faire une API qu'on peut appeler avec ?first_name=, ?last_name=, ?birthdate=, bref, pas de ?q= magique. Donc ce n'est pas la même chose, mais il faudrait peut-être éviter d'inventer deux trucs en parallèle. (et puis de ce que je lis plus le ticket a avancé plus le ?q= est devenu magique, ce qui n'était pas forcément l'objectif initial ; est-ce qu'il y a au moins un cas d'usage derrière ?)

En tout cas je n'ai jamais joué avec tout ça et pour rebondir sur le plan final, j'ai quelques questions :

Benjamin Dauvergne a écrit :

> plutôt que d'attribuer un comportement particulier à nom/prénom, construire un champ display_name comme dans w.c.s. à partir de nom/prénom (ce sera configurable comme w.c.s.) ce champ est particulier et indexé via trigram/unaccent/lower (zoo-like);

Je ne suis pas sûr d'avoir trouvé où c'était fait dans w.c.s. et encore moins où c'était configurable. Sinon, ça veut dire quoi « construire un champ » « particulier » ? Est-ce que c'est un CharField normal qui est mis à jour avec la concaténation nom prénom à chaque save(), et donc du code python, ou c'est un truc plus bas niveau ?
Ensuite pour l'index, mes recherches disent qu'il faut, à partir de l'hypothétique CharField display name, un truc genre :

```
CREATE INDEX display_name_idx ON user USING gin(lower(display_name)) gin_trgm_ops);
```

Tout du moins en django < 2, parce qu'on ne peut pas encore préciser d'opération avec Index de django.models.indexes.
Ça paraît aller dans le bon sens ?
Par contre pour ajouter unaccent ça a pas l'air gagné, cf
https://stackoverflow.com/questions/11005036/does-postgresql-support-accent-insensitive-collations.

- pour les tous les autres champs marqué "Pris en compte dans les recherches", créer une colonne fts comme dans w.c.s et l'alimenté, la recherche full text se fera dessus.

Même question, la colonne fts c'est un CharField où on agglutine tous les champs marqué searchable=True lors d'un save() ? Là aussi il faut créer un index ?

#11 - 05 avril 2022 15:39 - Benjamin Dauvergne

- Assigné à mis à Valentin Deniaud

J'ai l'impression qu'on a tout traité dans #46424.

#12 - 05 avril 2022 17:01 - Valentin Deniaud

- Statut changé de Nouveau à Fermé